대한민국특허청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 10-2003-0007728
Application Number

출 원 년 월 일 : 2003년 02월 07일
Date of Application FEB 07, 2003

출 원 인 : 삼성전자주식회사
Applicant(s) SAMSUNG ELECTRONICS CO., LTD.

2003 년 12 월 30 일

특 허 청
COMMISSIONER

【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0003
【제출일자】	2003.02.07
【발명의 명칭】	자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템 및 방법
【발명의 영문명칭】	SYSTEM AND METHOD OF SHORTENING CLASS LOADING PROCESS IN JAVA PROGRAM
【출원인】	
【명칭】	삼성전자 주식회사
【출원인코드】	1-1998-104271-3
【대리인】	
【성명】	김동진
【대리인코드】	9-1999-000041-4
【포괄위임등록번호】	2002-007585-8
【발명자】	
【성명의 국문표기】	송효정
【성명의 영문표기】	SONG,Hyo Jung
【주민등록번호】	691029-2041313
【우편번호】	120-091
【주소】	서울특별시 서대문구 홍제1동 367-22
【국적】	KR
【발명자】	
【성명의 국문표기】	박정규
【성명의 영문표기】	PARK,Jung Gyu
【주민등록번호】	730228-1799810
【우편번호】	130-034
【주소】	서울특별시 동대문구 답십리4동 1-424
【국적】	KR
【발명자】	
【성명의 국문표기】	최지영
【성명의 영문표기】	CHOI,Ji Young

【주민등록번호】	700418-2149521
【우편번호】	135-110
【주소】	서울특별시 강남구 압구정동 한양아파트 8-1102
【국적】	KR
【심사청구】	청구
【취지】	특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인 김동진 (인)
【수수료】	
【기본출원료】	16 면 29,000 원
【가산출원료】	0 면 0 원
【우선권주장료】	0 건 0 원
【심사청구료】	10 항 429,000 원
【합계】	458,000 원
【첨부서류】	1. 요약서·명세서(도면)_1통

【요약서】

【요약】

본 발명은 보조 기억장치로부터 자바 프로그램의 클래스 파일을 로딩하고, 링킹 과정 및 초기화 과정을 수행하여 런타임 데이터를 생성하는 클래스 로더부와, 상기 클래스 로더부에서 생성한 런타임 데이터를 엑세스 가능한 상태로 유지하는 제1 메모리부와, 상기 제1 메모리부에 엑세스 가능한 상태로 로딩된 런타임 데이터를 이미지 형태로 저장하는 제2 메모리부와, 상기 클래스 로더부의 요청에 따라 제2 메모리부에 이미지 형태로 저장된 런타임 데이터를 제1 메모리부로 불러오는 런타임 데이터 검색부 및 상기 제1 메모리부에 엑세스 상태로 로딩된 런타임 데이터들을 실행시키는 실행부를 포함하는 것을 특징으로 한다.

【대표도】

도 1

【색인어】

자바 프로그램, 자바 가상 머신, 클래스 로딩

【명세서】

【발명의 명칭】

자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템 및 방법{SYSTEM AND METHOD OF SHORTENING CLASS LOADING PROCESS IN JAVA PROGRAM}

【도면의 간단한 설명】

도 1은 본 발명의 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템을 개략적으로 나타낸 블럭도.

도 2는 본 발명의 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법을 개략적으로 나타낸 플로우챠트.

도 3은 상기 도 2의 런타임 데이터 생성 단계를 상세하게 나타낸 플로우챠트.

< 도면의 주요부분에 대한 부호의 설명 >

100 : 클래스 로더부 200 : 제1 메모리부

300 : 런타임 데이터 검색부 400 : 제2 메모리부

500 : 실행부 600 : 가비지 컬렉터부

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<8> 본 발명은 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템 및 방법에 관한 것으로서, 특히 자바 프로그램의 클래스 로딩 과정 수행시 생성된 런타임 데이터를 이미지 형태로 저장하고, 이후에 자바 프로그램을 시행할 때 저장된 이미지 형태의 런타임 데이터를 불러

와 실행 시킴으로써 클래스 로딩 시간을 줄일 수 있는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템 및 방법에 관한 것이다.

<9> 최근 들어, 무선단말기의 사용이 확산됨에 따라서 무선단말기에서 작동되는 자바(Java) 프로그램(예를 들어, 게임, 메신저 등)의 크기가 점점 커지고 있는 추세이다.

<10> 그런데, 자바 프로그램을 실행하려면 자바 가상 머신(Java Virtual Machine : JVM)에서 클래스 로딩(class loading) 과정이 수행되어 자바 프로그램이 실행된다. 그러나, 상기 클래스 로딩(class loading)과정은 로딩(loading), 링킹(linking) 및 초기화 (initialization) 등의 과정을 거쳐야 하며, 상기 링킹(linking)과정은 검증(verification), 예비(preparation) 및 결정(resolution) 등의 과정들을 포함 함으로써, 클래스 로딩(class loading)과정을 수행하는데 많은 시간이 소모되는 문제점이 있다.

<11> 특히, 클래스 로딩 과정 중 검증(verification) 과정은 동일한 바이트 코드(byte code)에 대하여 자바 프로그램을 처음 실행할 때만 수행하면 되지만, 현재의 클래스 로딩 과정은 자바 프로그램을 실행할 때 마다 검증 과정을 수행하게 설계되어 있다. 따라서, 기존의 클래스 로딩 과정은 무선 단말기와 같은 저 성능의 CPU와 저 용량의 배터리를 가진 시스템에서 응답 시간(response time)을 증가시키고, 배터리 소모를 많이 일으키는 문제점이 있다.

<12> 또한, 자바 프로그램의 크기가 커 질수록 프로그램의 로딩에 걸리는 시간이 증가하여 무선 단말기와 같은 저 성능의 CPU와 저 용량의 배터리를 가진 시스템에서 응답 시간을 증가시키고, 배터리 소모를 많이 일으키는 문제점이 있다.

【발명이 이루고자 하는 기술적 과제】

<13> 본 발명은 상기한 문제점을 해결하기 위하여 안출된 것으로서, 본 발명의 목적은 자바 프로그램의 클래스 로딩 과정 수행 후 생성되는 런타임 데이터를 이미지 형태로 저장함으로써, 다음 자바 프로그램 실행시 저장된 이미지 형태의 런타임 데이터를 불러와 실행 시킴으로써 클래스 로딩 시간을 줄일 수 있는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템 및 방법을 제공하는 것이다.

<14> 본 발명의 다른 목적은 핸드폰과 같이 저 성능 CPU와 저 용량 메모리를 가진 기기에서 자바 프로그램의 처리 속도를 향상시킴으로써 사용자에 대한 응답 시간을 감소 및 배터리 소모량을 줄일 수 있는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템 및 방법을 제공하는 것이다.

【발명의 구성 및 작용】

<15> 상기 목적을 달성하기 위하여 본 발명은, 보조 기억장치로부터 자바 프로그램의 클래스 파일을 로딩하고, 링킹 과정 및 초기화 과정을 수행하여 런타임 데이터를 생성하는 클래스 로더부와, 상기 클래스 로더부에서 생성한 런타임 데이터를 엑세스 가능한 상태로 유지하는 제1 메모리부와, 상기 제1 메모리부에 엑세스 가능한 상태로 로딩된 런타임 데이터를 이미지 형태로 저장하는 제2 메모리부와, 상기 클래스 로더부의 요청에 따라 제2 메모리부에 이미지 형태로 저장된 런타임 데이터를 제1 메모리부로 불러오는 런타임 데이터 검색부 및 상기 제1 메모리부에 엑세스 상태로 로딩된 런타임 데이터들을 실행시키는 실행부를 포함하는 것을 특징으로 한다.

<16> 이하, 첨부한 도면들을 참조로 본 발명의 바람직한 실시예를 상세히 설명한다.

<17> 도 1은 본 발명의 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템을 개략적으로 나타낸 블럭도로서, 클래스 로더부(class loader)(100), 제1 메모리부(200), 런타임 데이터 검색부(300), 제2 메모리부(400), 실행부(500) 및 가비지 컬렉터부(garbage collector)(600)로 구성된다.

<18> 클래스 로더부(100)는 보조 기억장치로부터 자바 프로그램의 클래스 파일을 로딩하고, 링킹 과정 및 초기화 과정을 수행하여 런타임 데이터를 생성한다. 여기서, 상기 로딩 과정이란 보조기억장치에 위치한 클래스 파일들을 자바 가상 머신(Java Virtual Machine : JVM)으로 전달하는 과정을 말하고, 상기 링킹 과정은 상기 로딩된 클래스 파일이 자바 가상머신에 의해서 수행될 수 있는 상태로 만드는 과정을 말하는 것으로, 검증, 예비 및 결정 등의 과정들을 포함한다. 상기 런타임 데이터(runtime data)는 제1 메모리부(200)에 로딩되어 자바 프로그램을 실행시키는 데이터로서, constant pool, method table, field table 등으로 이해될 수 있다.

<19> 제1 메모리부(200)는 상기 클래스 로더부(100)에서 생성한 런타임 데이터를 엑세스 가능한 상태로 유지한다. 즉, 상기 클래스 로더부(100)에서 생성한 런타임 데이터를 소정의 메모리 영역에 저장하여 후술하는 실행부(500)가 상기 저장된 런타임 데이터를 엑세스 할 수 있도록 한다.

<20> 제2 메모리부(400)는 상기 제1 메모리부(200)에 엑세스 상태로 로딩된 런타임 데이터를 이미지 형태로 저장한다.

<21> 한편, 다른 실시예로 상기 제1 메모리부(200)와 제2 메모리부(400)는 물리적으로 1개의 메모리부로 구성될 수 있다.

<22> 런타임 데이터 검색부(300)는 상기 클래스 로더부(100)의 요청에 따라 제2 메모리부(400)에 저장된 런타임 데이터들을 제1 메모리부(200)로 불러오며, 또한 클래스 로더부(100)에서 생성한 런타임 데이터를 제2 메모리부(400)에 이미지 형태로 저장한다. 그리고, 상기 런타임 데이터 검색부(300)는 LRU(Least Recently Used) 방식을 이용하여 제2 메모리부(400)에 이미지 형태로 저장된 런타임 데이터를 관리한다. 여기서, 상기 LRU 방식이란 저장된 데이터들 중 자주 사용되지 않는 데이터를 체크하여 잘 사용되지 않는 순으로 해당 데이터를 버리는 것이다.

<23> 실행부(500)는 상기 제1 메모리부(200)에 엑세스 상태로 로딩된 런타임 데이터들을 실행시킨다.

<24> 가비지 컬렉터부(600)는 상기 제1 메모리부(200)에서 사용되지 않는 공간들을 모아서 재 사용할 수 있도록 하며, 이로써 상기 제1 메모리부(200)의 사용 공간을 확보해 준다.

<25> 도 2는 본 발명의 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법을 개략적으로 나타낸 플로우챠트 이다.

<26> 먼저, 클래스 로더부(100)가 런타임 데이터 검색부(300)에 자바 프로그램의 실행시 필요한 런타임 데이터를 요청하면(S100), 상기 런타임 데이터 검색부(300)는 제2 메모리부(400)에 런타임 데이터가 존재하는지를 검색한다(S110).

<27> 상기 제2 메모리부(400)에서 해당 런타임 데이터가 검색되면, 상기 검색된 런타임 데이터는 제1 메모리부(200)로 전송되고(S125), 상기 제1 메모리부에 전송된 런타임 데이터는 실행부(500)에 의해 실행된다(S160). 여기서, 상기 제2 메모리부(400)에 저장된 런타임 데이터는

이미지 형태로 저장된 파일로써, 즉 이전 자바 프로그램 실행시 만들어진 런타임 데이터를 이미지 형태로 저장해 놓은 파일을 말한다.

<28> 한편, 본원 발명에서는 이미 생성되어 제2 메모리부(400)에 저장된 런타임 데이터를 제1 메모리부(200)에 로딩하여 실행만 시키면되므로, 자바 프로그램을 실행시킬 때마다 런타임 데이터를 생성하지 않아도 되며, 이로써 런타임 데이터 생성에 소요되는 복잡한 로딩 과정을 생략할 수 있어 클래스 로딩 시간을 줄일 수 있다.

<29> 한편, 상기 런타임 데이터 검색부(300)가 제2 메모리부(400)를 검색한 결과 런타임 데이터가 존재하지 않은 경우, 상기 클래스 로더부(100)는 자바 프로그램의 실행시 필요한 런타임 데이터를 생성한다(S130).

<30> 상기 런타임 데이터를 생성하는 방법을 살펴보면, 먼저 보조 기억장치로부터 자바 프로그램의 클래스 파일을 로딩하고(S132), 상기 로딩된 클래스 파일을 링킹 과정 및 초기화 과정을 수행하여 런타임 데이터를 생성한다(S134 내지 S138). 여기서, 상기 로딩이란 보조기억장치에 위치한 클래스 파일들을 자바 가상 머신으로 전달하는 과정을 말하고, 상기 클래스 파일 링킹 과정은 로딩된 클래스 파일, 즉 상기 로딩된 클래스 파일이 자바 가상 머신에 의해서 수행될 수 있는 상태로 만드는 과정을 말한다. 더 상세하게는, 상기 로딩된 클래스 파일이 올바른 클래스 포맷을 가지고 있는지를 검증하는 검증 과정과, 메모리 영역을 할당하는 예비 과정 및 상기 클래스 파일을 실행 가능하도록 변환하는 결정 과정을 포함한다.

<31> 그 다음, 상기 링킹 과정 수행 후 클래스 파일을 초기화하여 런타임 데이터를 생성한다. 상기 생성된 런타임 데이터는 런타임 데이터 검색부(300)에 의해 제2 메모리부(400)에 이미지 형태로 저장된다(S140). 여기서, 상기 제2 메모리부(400)에 저장된 이미지 형태의 데이터들은

런타임 데이터 검색부(300)에 의해 LRU 방식으로 관리된다. 즉, 제2 메모리부(400)의 저장 영역이 한정되어 있기 때문에 LUR 방식을 적용하여 데이터들을 관리하는 것이다.

<32> 이 후, 이미지 형태로 저장된 런타임 데이터는 런타임 데이터 검색부(300)에 의해 제1 메모리부(200)로 전송되고(S150), 상기 제1 메모리부(200)로 전송된 이미지 형태의 런타임 데이터는 실행부(500)에 의해 실행된다(S260). 여기서, 상기 제1 메모리부(200)에 데이터를 로딩할 공간이 부족할 경우, 가비지 컬렉터부(600)에서 상기 제1 메모리부(200)에서 사용되지 않는 공간들을 모아서 재 사용할 수 있도록 해줌으로써 상기 제1 메모리부(200)의 공간을 확보해 준다.

<33> 한편, 상기 단계 S140은 단계 S160의 실행 이후 수행될 수도 있다. 즉, 런타임 데이터가 생성되면(S130), 상기 생성된 런타임 데이터를 제1 메모리부(200)에 전송하고(S150), 상기 전송된 런타임 데이터를 실행시킨다(S160). 그 다음, 상기 자바 프로그램의 실행이 종료될 때 상기 생성된 런타임 데이터를 제2 메모리부(400)에 저장할 수도 있다.

<34> 이상에서 본 발명에 대하여 상세히 기술하였지만, 본 발명이 속하는 기술 분야에 있어서 통상의 지식을 가진 사람이라면, 첨부된 청구범위에 정의된 본 발명의 정신 및 범위를 벗어나지 않으면서 본 발명을 여러 가지로 변형 또는 변경하여 실시할 수 있음은 자명하며, 따라서 본 발명의 실시예에 따른 단순한 변경은 본 발명의 기술을 벗어날 수 없을 것이다.

【발명의 효과】

<35> 상기한 구성의 본 발명에 의하면, 자바 프로그램의 클래스 로딩 과정 수행 후 생성된 런타임 데이터를 이미지 형태로 저장함으로써, 다음 자바 프로그램 실행시 저장된 이미지 형태의 런타임 데이터를 불러와 실행시킴으로써 클래스 로딩의 복잡한 과정을 수행하지 않고 자바 프

로그램을 실행시킬 수 있어 자바 프로그램 실행시 클래스 로딩 시간을 줄일 수 있는 잇점이 있다.

<36> 또한, 저 성능 CPU와 저 용량 메모리를 가진 기기에서 자바 프로그램의 처리 속도를 향상시킴으로써 사용자에 대한 응답 시간을 감소시키고 배터리의 소모량을 줄일 수 있는 잇점이 있다.

【특허청구범위】

【청구항 1】

보조 기억장치로부터 자바 프로그램의 클래스 파일을 로딩하고, 링킹 과정 및 초기화 과정을 수행하여 런타임 데이터를 생성하는 클래스 로더부;

상기 클래스 로더부에서 생성한 런타임 데이터를 엑세스 가능한 상태로 유지하는 제1 메모리부;

상기 제1 메모리부에 엑세스 가능한 상태로 로딩된 런타임 데이터를 이미지 형태로 저장하는 제2 메모리부;

상기 클래스 로더부의 요청에 따라 제2 메모리부에 이미지 형태로 저장된 런타임 데이터를 제1 메모리부로 불러오는 런타임 데이터 검색부; 및

상기 제1 메모리부에 엑세스 상태로 로딩된 런타임 데이터들을 실행시키는 실행부를 포함하는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템.

【청구항 2】

제 1항에 있어서,

상기 제1 메모리부에서 사용되지 않는 공간들을 모아서 재 사용할 수 있도록 하는 가비지 컬렉터부를 더 포함하는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템.

【청구항 3】

제 1항에 있어서, 상기 런타임 데이터 검색부는,

상기 클래스 로더부에서 생성한 런타임 데이터를 제2 메모리부에 이미지 형태로 저장하는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템.

【청구항 4】

제 1항 또는 3항에 있어서, 상기 런타임 데이터 검색부는,

LRU 방식을 이용하여 상기 제2 메모리부에 이미지 형태로 저장된 런타임 데이터를 관리하는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 시스템.

【청구항 5】

클래스 로더부가 런타임 데이터 검색부에 자바 프로그램의 실행시 필요한 런타임 데이터를 요청하는 단계;

상기 런타임 데이터 검색부가 상기 요청받은 자바 프로그램에 대한 런타임 데이터를 검색하는 단계;

상기 검색된 해당 런타임 데이터를 제1 메모리부에 전송하는 단계; 및

상기 제1 메모리부로 전송된 런타임 데이터를 실행시키는 단계를 포함하는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법.

【청구항 6】

제 5항에 있어서,

상기 검색된 런타임 데이터는 제2 메모리부에 이미지 형태로 저장되어 있는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법.

【청구항 7】

제 6항에 있어서, 상기 제2 메모리부에 이미지 형태로 저장되어 있는 런타임 데이터는,

런타임 데이터 검색부에 의해 LRU 방식을 통해 관리되는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법.

【청구항 8】

제 5항에 있어서, 상기 요청받은 자바 프로그램에 대한 런타임 데이터 검색 결과 런타임 데이터가 존재하지 않을 경우에는,

보조 기억장치로부터 자바 프로그램의 클래스 파일을 로딩하는 단계;

상기 로딩된 클래스 파일을 링킹 과정 및 초기화 과정을 수행하여 런타임 데이터를 생성하는 단계;

상기 생성된 런타임 데이터를 이미지 형태로 저장하는 단계;

상기 이미지로 저장된 런타임 데이터를 제1 메모리부로 전송하는 단계를 더 포함하는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법.

【청구항 9】

제 8항에 있어서, 상기 생성된 런타임 데이터를 이미지 형태로 저장하는 단계는,

상기 제1 메모리부로 전송된 런타임 데이터를 실행시키는 단계 이후 수행되는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법.

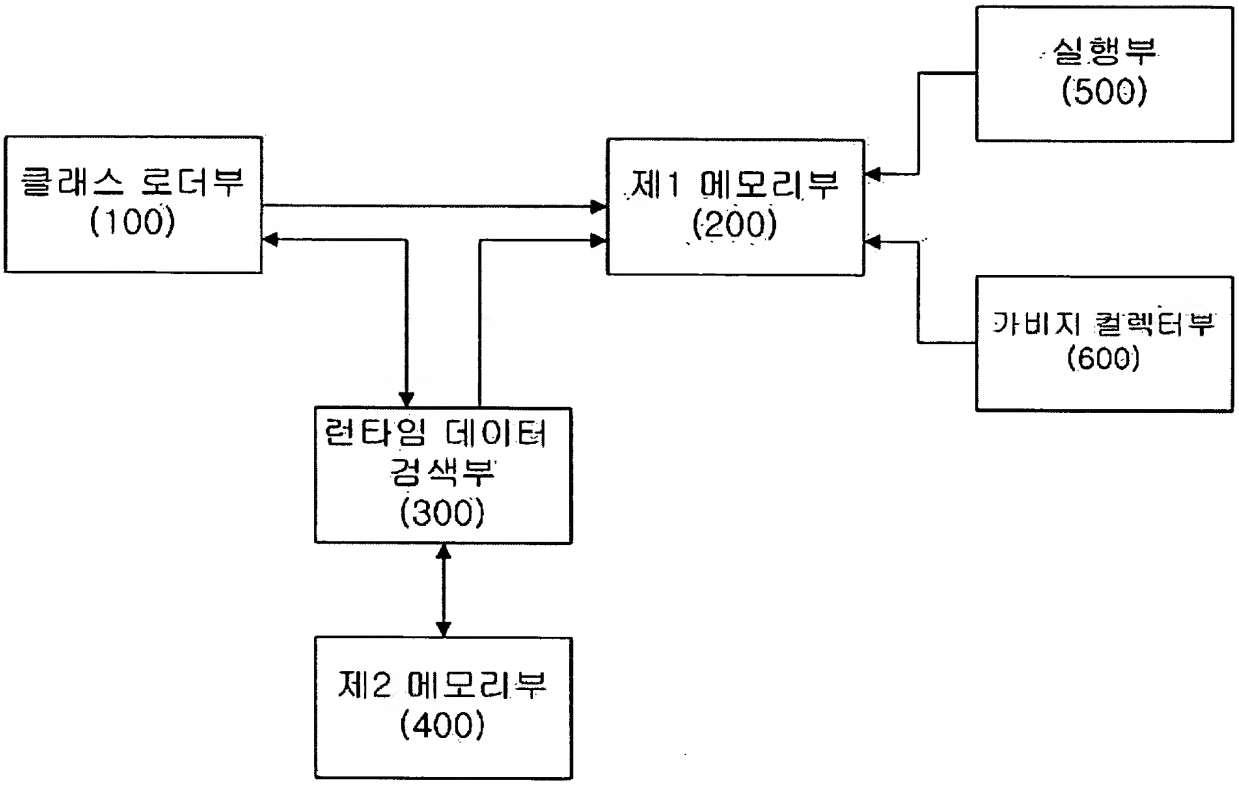
【청구항 10】

· 제 8항에 있어서, 상기 저장된 이미지 형태의 런타임 데이터는,

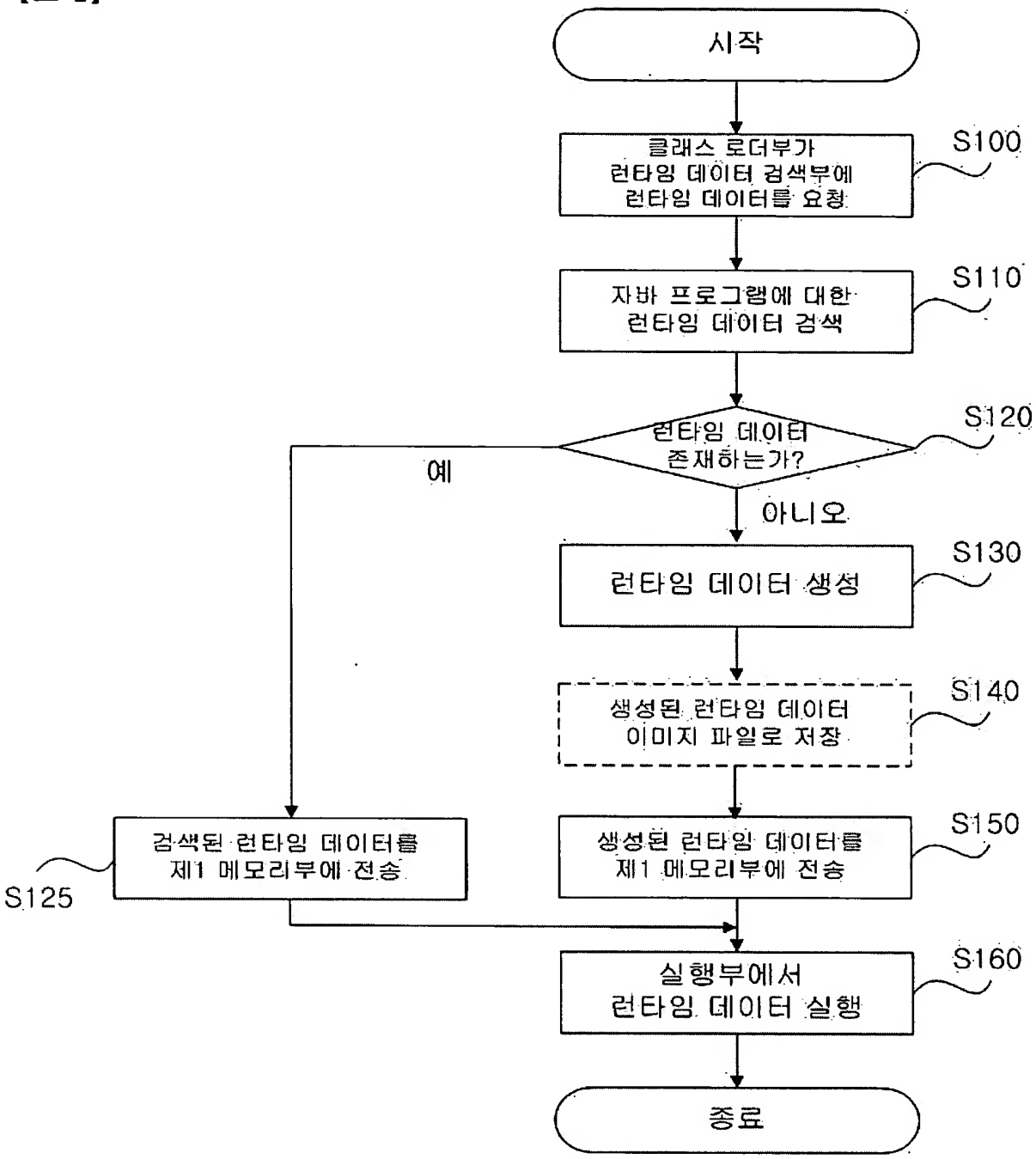
런타임 데이터 검색부에 의해 LRU 방식을 통해 관리되는 것을 특징으로 하는 자바 프로그램에서 클래스 로딩 과정을 단축시키는 방법.

【도면】

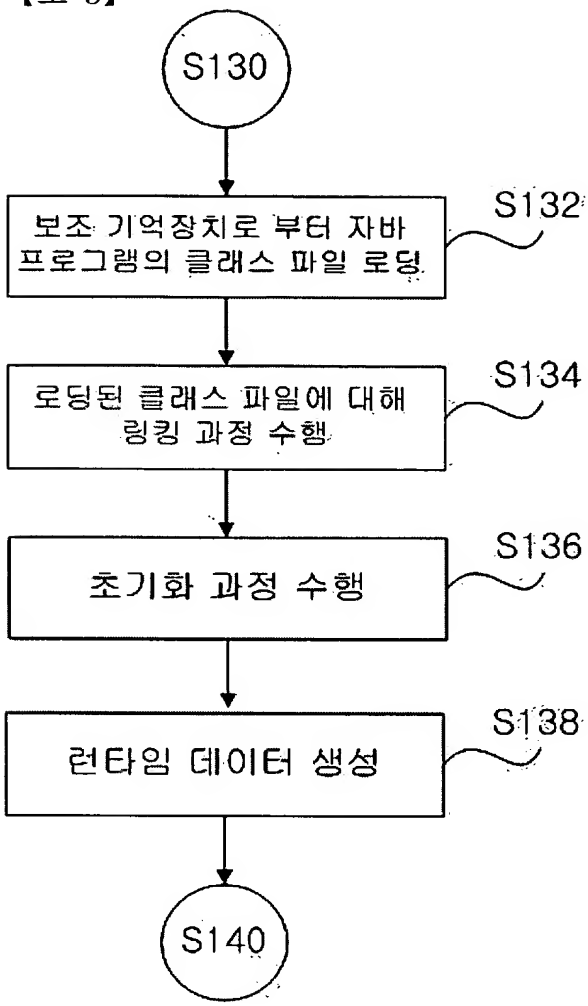
【도 1】


실행부
(500)
클래스 로더부
(100)
제1 메모리부
(200)
가비지 컬렉터부
(600)
런타임 데이터
검색부
(300)
제2 메모리부
(400)

【도 2】


시작
클래스 로더부가 런타임 데이터 검색부에 런타임 데이터를 요청
S100
자바 프로그램에 대한 런타임 데이터 검색
S110
런타임 데이터 존재하는가?
S120
예
아니오
런타임 데이터 생성
S130
생성된 런타임 데이터 이미지 파일로 저장
S140
검색된 런타임 데이터를 제1 메모리부에 전송
S125
생성된 런타임 데이터를 제1 메모리부에 전송
S150
실행부에서 런타임 데이터 실행
S160
종료

【도 3】


S130
보조 기억장치로 부터 자바 프로그램의 클래스 파일 로딩
S132
로딩된 클래스 파일에 대해 링킹 과정 수행
S134
초기화 과정 수행
S136
런타임 데이터 생성
S138
S140